Arbeiten mit json2conf innerhalb Visual Studio 2015

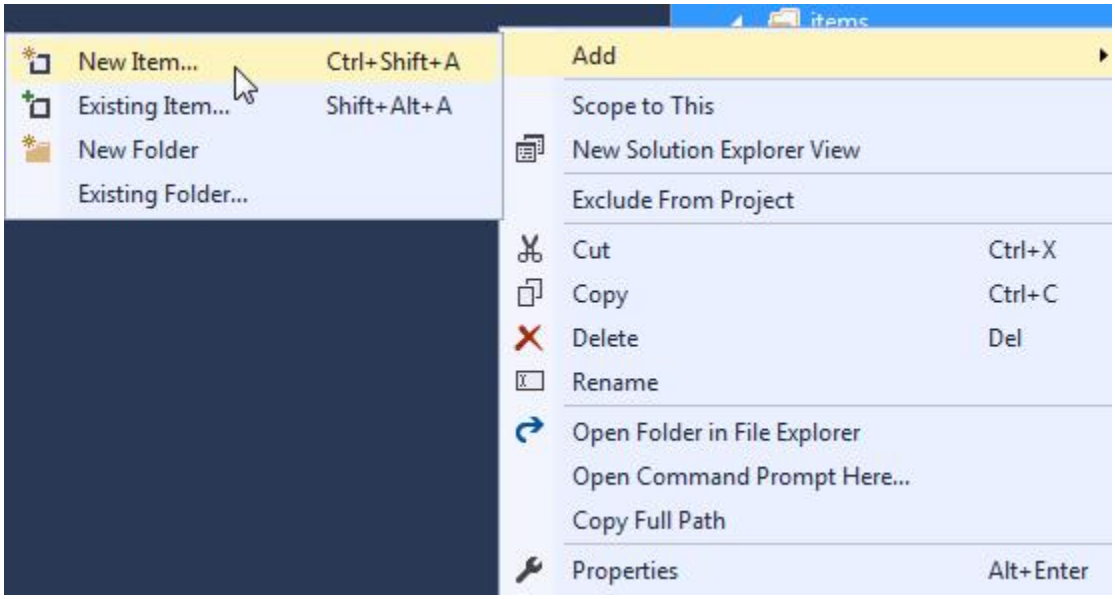
Hier wird anhand eines bebilderten Beispiels eine Erstellung eines einfachen conf-Fiels für smarthome.py erklärt.

Ziel ist es, die Arbeitsweise mit Hilfe von Visual Studio in Zusammenarbeit mit json2conf aufzuzeigen und eine erste Einstiegshilfe zu geben.

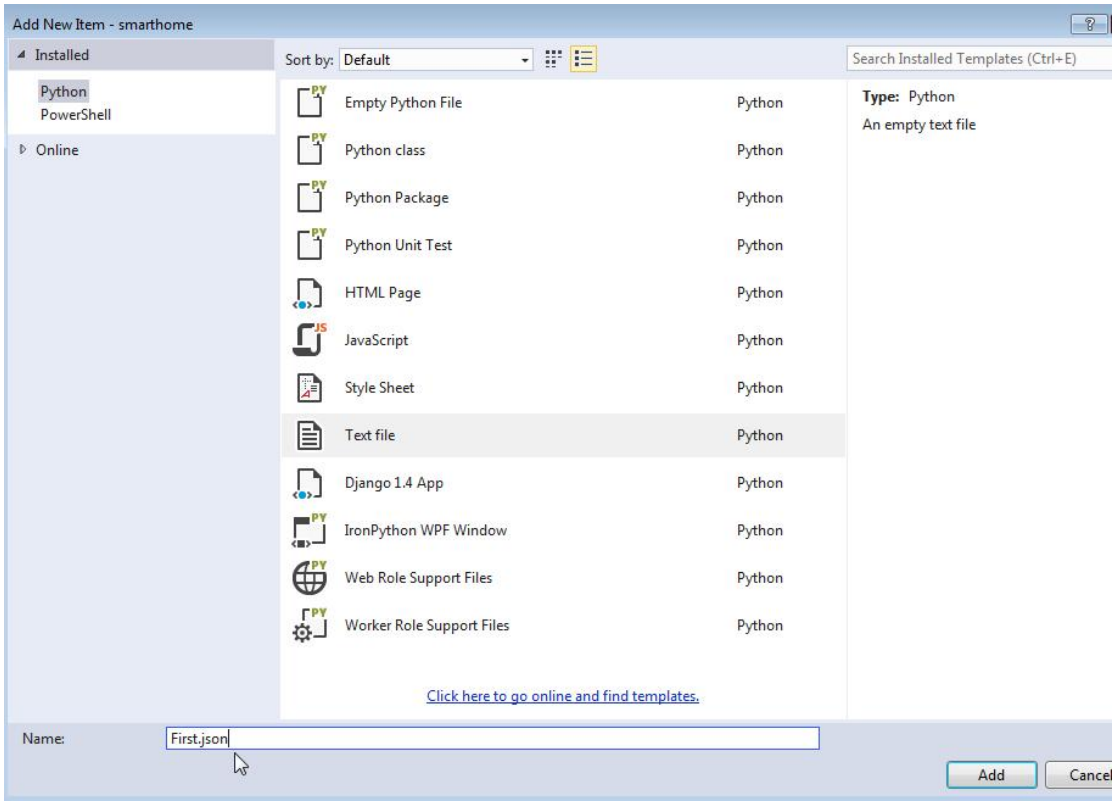
Voraussetzung: Das Dokument InitialSetup.pdf wurde durchgearbeitet und alle darin enthaltenen Punkte erfolgreich umgesetzt.

Neues Item-File erstellen

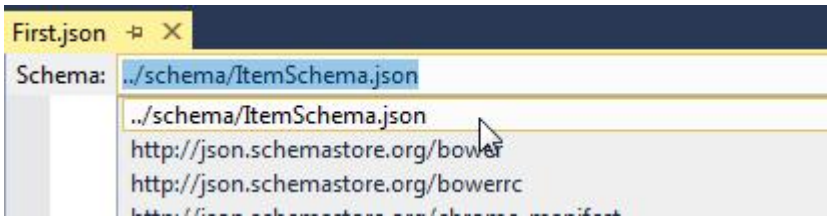
Als erstes wird ein neues Item-File First.json erstellt. Dazu mit der rechten Maustaste auf smarthome/items clicken und Add->New Item... auswählen.



In dem Folgedialog bitte auf TestFile clicken und unten im Eingabefeld "First.json" tippen, anschließend auf "Add" clicken.



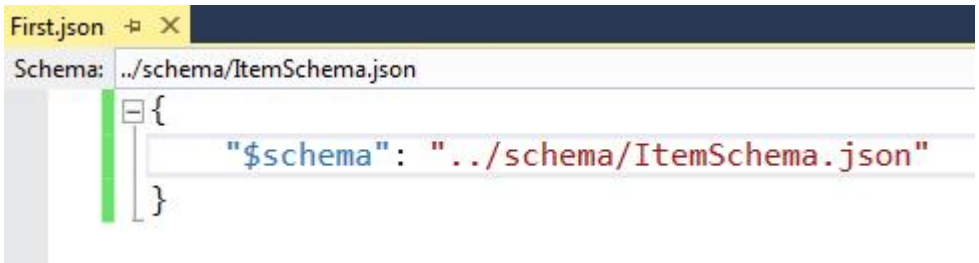
Die Datei First.json wird auf der rechten Seite unter items hinzugefügt und gleichzeitig im Editor geöffnet. Hier in der oberen Dropdown bitte ..\schema \ItemSchema.json auswählen.



Nun bitte exakt folgende Tasten tippen (und dabei beobachten, was auf dem Bildschirm passiert):

{ <enter> " <tab> <tab> <Strg+s>

Im Editor sollte man jetzt folgendes sehen:



Damit ist ein neues Item-File erzeugt.

Neues Item erstellen

Als nächstes wollen wir ein neues Item erstellen. Dazu den Cursor auf das Ende der Zeile setzen, hinter der das neue Item erzeugt werden soll (im aktuellen File wäre das hinter ...ItemSchema.json".

Mit folgende (wenigen) Tastendrücken erstellt man ein korrektes Item mit knx-Anbindung:

<komma> <enter> "First : <enter> <enter> ty : <tab> <komma> <enter>

na : Mein erstes Item <komma> <enter>

dp : <tab> <komma> <enter>

kn : 1/1/1 <komma> <enter>

se : 1/2/1

<Strg+s>

Jetzt sollte im Editor folgendes stehen:

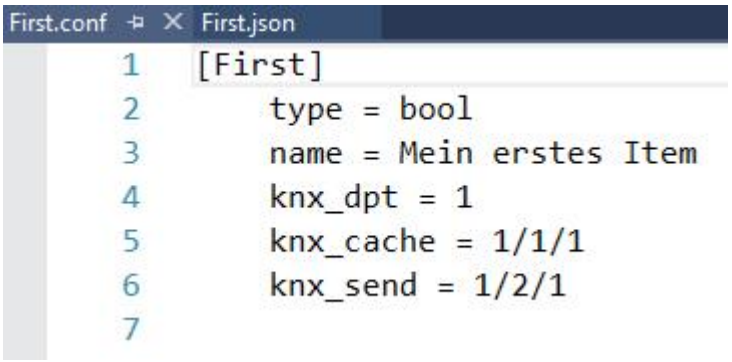
Das mit den Tastendrücken soll nur zeigen, wie die code completion funktioniert. Man muss auf jeden Fall wesentlich weniger tippen, als man denkt, um korrektes JSON zu erhalten.

Konvertieren in ein .conf-File

Nun kann man einfach F5 drücken. Es erscheint kurz ein schwarzes Fenster. Sobald das verschwindet, gibt es rechts im Solution Explorer unter

smarthome/items ein neues File First.conf, das kann man mit einem Doppelclick öffnen.

Die Konvertierung sollte so aussehen:

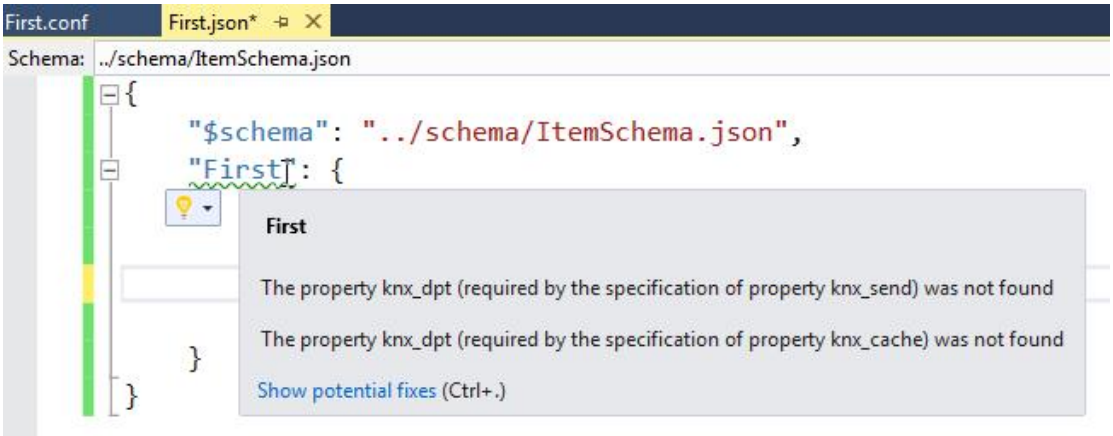


Das war ein simpler Roundtrip. Im folgenden werden noch die Fehlermeldungen und Warnungen besprochen.

Fehlermeldungen im Editor

Um einen Fehler zu provozieren, löschen wir im Editor in First.json die Zeile mit knx_dpt, indem wir den Cursor auf die Zeile setzen und <Strg+x> drücken.

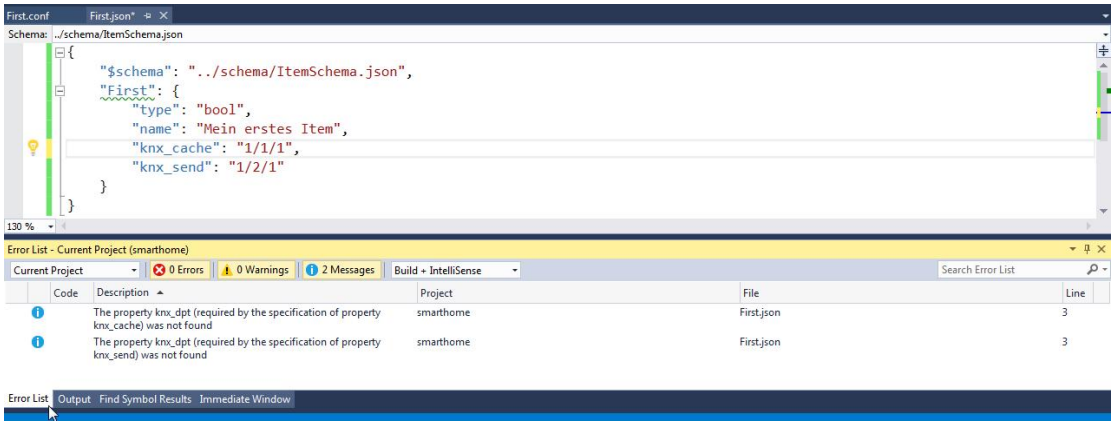
Sofort wird das "First" grün unterstrichen und sobald man mit der Maus drauf geht, bekommt man folgende Info:



Die Meldung besagt nichts anderes, als dass man knx_dpt braucht, weil man

knx_send bzw. knx_cache benutzt hat.

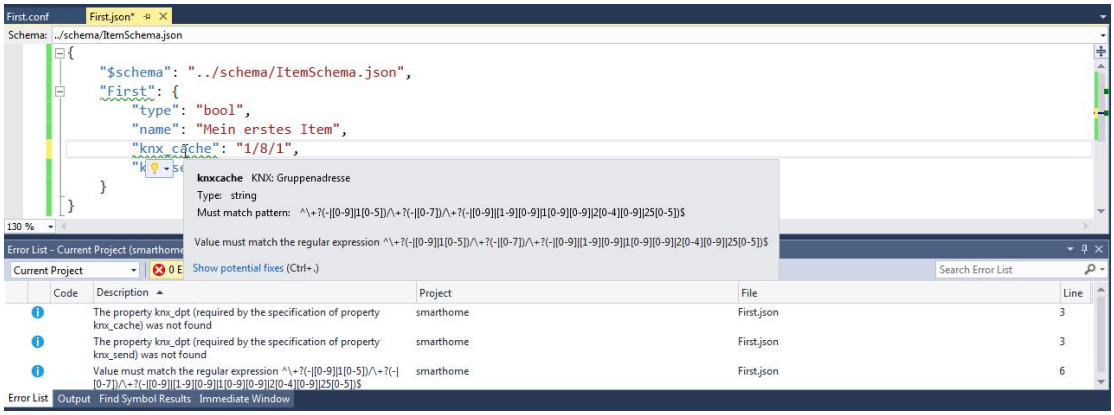
Diese Meldung erscheint auch in der Error List, das ist ein Tab am unteren Rand des Visual Studio Fensters.



Technisch gesehen sind das 2 Meldungen: Einmal fehlt knx_dpt für knx_send, einmal für knx_cache.

Ein Doppelclick auf eine Meldung bringt einen zur Fehlerstelle (bitte ausprobieren).

Ein anderes Beispiel: Bei knx_cache die GA auf 1/8/1 ändern, dann kommt:



Mag kryptisch aussehen, heißt aber, dass die GA nicht dem regulären Ausdruck genügt, der für GA hinterlegt ist: Hauptgruppe 0-15, Mittlegruppe 0-7, Untergruppe 0-255.

Es gibt eine Reihe von Bedingungen für die Korrektheit des JSON-Dokuments, die durch das zugrundeliegende JSON-Schema überprüft werden. Da sich der Konverter auf korrekte JSON-Dokumente verläßt (also welche, die dem Schema genügen), er aber seinerseits (noch) keine eigene Schema-Verifikation durchführt, kann es zu Laufzeitfehlern kommen, wenn er mit einem inkorrekten JSON-Schema aufgerufen wird.

Folglich gilt die Regel: Erst F5 (ausführen) drücken, wenn alle Meldungen in der Error List korrigiert sind.

Bitte jetzt das Item wieder korrigieren (knx_dpt: 1 und 1/1/1 für die GA)

Fehlermeldungen durch den Konverter

Selbst wenn das JSON formal korrekt ist, können Fehler drin stecken, die verhindern, das sh.py korrekt arbeitet. Da der Konverter soweiso alle Items lesen und konvertieren muss, sind auch noch weitere Checks eingebaut.

In erster Linie werden derzeit alle verwendeten Item-Referenzen überprüft. Bei nicht vorhandenen Items wird eine entsprechende Warnung ausgegeben.

Erstmal ein Beispiel:

Wir erweitern unser Item um ein Unteritem, so dass das Ganze am Ende so aussieht:

```
{
    "$schema": "../schema/ItemSchema.json",
    "First": {
        "type": "bool",
        "name": "Mein erstes Item",
        "knx_dpt": "1",
        "knx_cache": "1/1/1",
        "knx_send": "1/2/1",
        "eval": "not value",
        "eval_trigger": "first.sub",
        "Sub": {
            "type": "bool",
            "name": "Trigger",
            "knx_dpt": "1",
            "knx_cache": "7/7/7"
        }
    }
}
```

(Bitte eintippen, als Übung für den Editor).

Auf den ersten Blick sieht das gut aus. Nun bitte F5 drücken und in der unteren Tab-Leiste auf Output clicken (ist neben Error List).

Da steht die Meldung:

```
Item not found: first.sub, referenced in File: items\First.json, Path:
First.eval_trigger
```

Daran sieht man, dass man ja nicht first.sub, sondern First.Sub hätte schreiben müssen. Hätte man das an sh.py geschickt, wäre das auch im Log zu finden gewesen, allerdings erst nach dem Start von sh.py. So weiß man den Fehler bereits vorher.

Die Überprüfung von Item-Referenzen ist auch sehr hilfreich, wenn man relative Referenzen verwendet. Wir ersetzen eval_trigger durch

```
        "eval_trigger": "~.sub",
```

(Das besagt: Nehme das aktuelle Item und von da aus das Unteritem sub).

Wenn man jetzt F5 drückt, bekommt man folgende Meldungen:

```
Item not found: ~.sub, referenced in File: items\First.json, Path:
First.eval_trigger

Item not found: First.sub, referenced in File: items\First.json, Path:
First.eval_trigger
```

Dass es 2 Meldungen sind, ist technisch bedingt. Die erste zeigt, wie die Referenz notiert wurde (also relativ mit ~.sub), die 2. Meldung zeigt, wonach gesucht wurde (First.sub).

Jetzt korrigieren wir das zu

```
        "eval_trigger": "~.Sub",
```

und mit F5 sehen wir, dass es keine Meldung mehr gibt. Im First.conf sehen wir auch, dass ~.Sub zu First.Sub aufgelöst wurde.

Die Fehlerliste wird auch in smarthome/items/GA/Messages.txt geschrieben. Dort gibt es auch die in der Anleitung beschriebenen Files, die alle GA-Referenzen beinhalten.

Viel Spaß mit dem Konverter!

Gruß, Waldemar